Taller de Iniciación a Arduino

Presentación

Este documento sirve como apoyo visual y repositorio de código para las tres sesiones del taller.

SESIÓN 1: Introducción, IDE y Control de LEDs

1. El Ecosistema Arduino

A. La Placa Arduino Uno

Características y Funcionamiento Básico de la Placa Arduino Uno

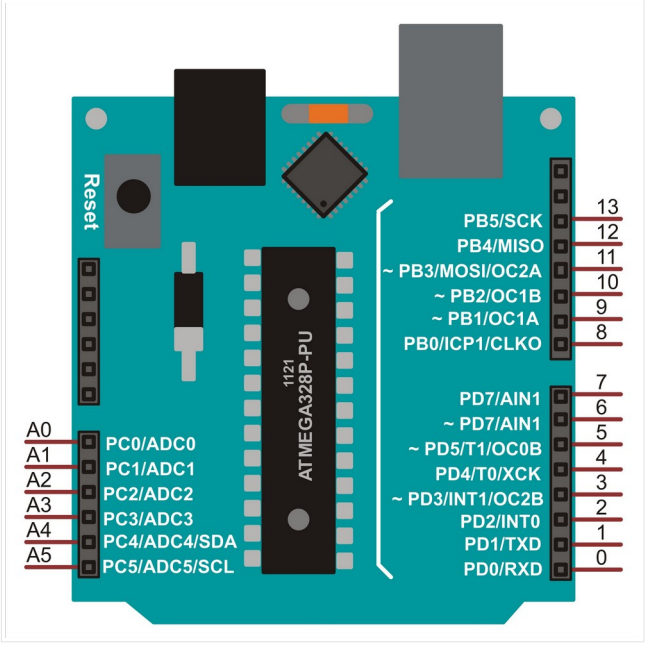
Este documento resume los componentes esenciales de la placa Arduino Uno y explica la estructura fundamental de su código de programación.

1. Características Básicas de la Placa Arduino Uno

La Arduino Uno es la placa más popular para principiantes. Está basada en el microcontrolador **ATmega328P** de Atmel.

Característica	Valor	Notas
Microcontrolador	ATmega328P	El cerebro de la placa.
Voltaje de Operación	5V	El voltaje al que trabajan los pines y la lógica de la placa.
Voltaje de Entrada (Recomendado)	7V a 12V	Para alimentar la placa a través del conector de barril.
Memoria Flash	32 KB	Para almacenar el código del programa (.ino).
Pines Digitales I/O	14	Pueden ser configurados como entradas o salidas.
Pines PWM (Salida	6	Son un subconjunto de los pines

Analógica)		digitales (3, 5, 6, 9, 10, 11).
Pines de Entrada Analógica	6 (A0 a A5)	Para medir voltajes variables (sensores).
Velocidad de Reloj	16 MHz	Frecuencia de operación del microcontrolador.



Shutterstock Explorar

2. Tipos de Conexión y Pines

La placa Arduino Uno organiza sus pines en varias categorías, cada una con un propósito específico:

A. Pines de Alimentación (Power)

- **5V:** Proporciona un voltaje regulado de 5 voltios.
- **3.3V:** Proporciona un voltaje regulado de 3.3 voltios.
- **GND (Tierra):** Referencia de voltaje ($0V$). Todo circuito debe estar conectado a tierra.
- **VIN:** Voltaje de entrada de la fuente externa (ej. batería o adaptador de pared).

B. Pines Digitales (Digital I/O)

Los 14 pines numerados (0 a 13) se utilizan para entradas y salidas binarias (ON u OFF).

- **Salida (OUTPUT):** Permiten enviar $5V$ (HIGH) o $0V$ (LOW) para controlar dispositivos (LEDs, relés, etc.).

- **Entrada (INPUT):** Permiten leer si un dispositivo está enviando $5V$ (HIGH) o $0V$ (LOW) (botones, interruptores).
- **PWM (~):** 6 de estos pines (3, 5, 6, 9, 10, 11) pueden generar Modulación por Ancho de Pulso, permitiendo simular salidas analógicas (controlar brillo de LEDs o velocidad de motores).

C. Pines de Entrada Analógica (Analog In)

Los pines etiquetados de **A0 a A5** se utilizan para leer voltajes que varían continuamente, como los provenientes de sensores de temperatura, luz o potenciómetros. Convierten un voltaje de $0V$ a $5V$ en un valor digital de 0 a 1023 (Resolución de 10 bits).

D. Pines de Comunicación Serie

- **Serial (UART):** Los pines **0 (RX)** y **1 (TX)** se utilizan para la comunicación Serial con el ordenador (a través del Monitor Serie) o con otros dispositivos.
- **I2C:** Los pines **A4 (SDA)** y **A5 (SCL)**, que también son pines analógicos, se utilizan para la comunicación con ciertos sensores y módulos, como el BMP280, utilizando el protocolo I2C.

3. Funcionamiento Básico de un Script de Arduino

Un programa de Arduino, llamado *Sketch*, se escribe en el IDE (Entorno de Desarrollo Integrado) y consta de dos funciones obligatorias:

A. La Función `setup()`

Esta función se ejecuta **una sola vez** cuando la placa se enciende o se reinicia. Su propósito es configurar los parámetros iniciales del programa:

1. **Configuración de Pines:** Establecer si cada pin digital se usará como entrada (`INPUT`) o salida (`OUTPUT`) mediante `pinMode()`.
2. **Inicialización de Comunicaciones:** Iniciar la comunicación Serial (`Serial.begin()`) o los protocolos (I2C) si se van a utilizar.
3. **Inicialización de Librerías:** Preparar cualquier componente externo que requiera librerías (como el BMP280).

B. La Función `loop()`

Esta función se ejecuta **repetidamente y de forma infinita** después de que `setup()` haya terminado. Es el corazón del programa.

1. **Lógica del Programa:** Contiene las instrucciones que definen el comportamiento de la placa, como leer un sensor, tomar una decisión basada en esa lectura y activar un actuador.
2. **Ejecución Continua:** Si la placa necesita parpadear un LED o leer datos de forma periódica, ese código reside en el `loop()`.

Ejemplo de Estructura de Código

El siguiente código, que usamos para el ejercicio de parpadeo (blink.ino), ilustra perfectamente esta estructura:

```
// Sección de Declaración de Variables Globales (fuera de las funciones)
const int LED_PIN = 13;

// 1. FUNCIÓN DE CONFIGURACIÓN
void setup() {
  // Configuración: solo se ejecuta una vez
  pinMode(LED_PIN, OUTPUT);
}

// 2. FUNCIÓN DE BUCLE PRINCIPAL
void loop() {
  // Lógica principal: se repite sin parar
  digitalWrite(LED_PIN, HIGH); // Enciende el LED
  delay(1000);                 // Espera 1 segundo
  digitalWrite(LED_PIN, LOW);  // Apaga el LED
  delay(1000);                 // Espera 1 segundo
}
```